IN THE CLAIMS:

1. (currently amended) A context switching system for a multi-thread execution pipeline loop having a pipeline latency, comprising:

a miss fulfillment first-in-first-out buffer (FIFO);

a context switch requesting subsystem configured to:

detect a device request from a thread executing within said multi-thread execution pipeline loop for access to a device having a fulfillment latency exceeding said pipeline latency, and

generate a context switch request for said thread; and

a context controller subsystem configured to receive said context switch request and, based thereon, store said thread in said miss fulfillment FIFO to prevent said thread from executing until said device request is fulfilled, said thread sequencing through said entire miss fulfillment FIFO before exiting therefrom at a rate having a period associated with said pipeline latency.

2. (currently amended) The context switching system as recited in Claim 1 wherein said context controller subsystem is further configured to allow a new thread to enter said multi-thread execution pipeline loop after storing said thread in said miss fulfillment ~~misfulfillment~~ FIFO.

3. (original) The context switching system as recited in Claim 1 wherein said context controller subsystem is further configured to allow other threads within said multi-thread execution pipeline loop to continue to execute while said thread is waiting for said device request to be fulfilled.

4. (previously presented) The context switching system as recited in Claim 1 wherein said context controller subsystem is further configured to:

store said thread in said miss fulfillment FIFO upon reaching an end position of said multi-thread execution pipeline loop, and

reinsert said thread into said multi-thread execution pipeline loop at a beginning position after said thread exits said miss fulfillment FIFO.

5. (previously presented) The context switching system as recited in Claim 1 wherein said thread is looped back to a beginning stage of said multi-thread execution pipeline loop when said thread reaches an end stage of said multi-thread execution pipeline loop and said thread has not finished processing.

6. (currently amended) The context switching system as recited in Claim 1 wherein said context controller subsystem is further configured to sequence said thread through said miss fulfillment FIFO at a rate equal ~~having a period substantially equivalent~~ to said pipeline latency of said multi-thread execution pipeline loop.

7. (original) The context switching system as recited in Claim 1 wherein said device request is a request to access external memory due to a cache miss status.

8. (currently amended) For use with a multi-thread execution pipeline loop having a pipeline latency, a method of operating a context switching system, comprising:

detecting a device request from a thread executing within said multi-thread execution pipeline loop for access to a device having a fulfillment latency exceeding said pipeline latency;

generating a context switch request for said thread when said thread issues said device request; and

receiving said context switch request and storing said thread based thereon in a miss fulfillment first-in-first-out buffer (FIFO) until said device request is fulfilled, said thread sequencing through said entire miss fulfillment FIFO before exiting therefrom at a rate having a

period associated with said pipeline latency.

9. (previously presented) The method as recited in Claim 8 further comprising allowing a new thread to enter said multi-thread execution pipeline loop after storing said thread in said miss fulfillment FIFO.

10. (original) The method as recited in Claim 8 further comprising allowing other threads within said multi-thread execution pipeline loop to continue to execute while said thread is waiting for said device request to be fulfilled.

11. (previously presented) The method as recited in Claim 8 further comprising:

storing said thread in said miss fulfillment FIFO upon reaching an end position of said multi-thread execution pipeline loop, and

reinserting said thread into said multi-thread execution pipeline loop at a beginning position after said thread exits said miss fulfillment FIFO.

12. (previously presented) The method as recited in Claim 8 further comprising looping said thread back to a beginning stage of said multi-thread execution pipeline loop when said thread reaches an end stage of said multi-thread execution pipeline loop and said thread has not finished processing.

13. (currently amended) The method as recited in Claim 8 further comprising sequencing said thread through said miss fulfillment FIFO at a rate equal ~~having a period substantially equivalent~~ to said pipeline latency of said multi-thread execution pipeline loop.

14. (original) The method as recited in Claim 8 wherein said device request is a request to access external memory due to a cache miss status.

15. (currently amended) A fast pattern processor that receives and processes protocol data units (PDUs), comprising:

a dynamic random access memory (DRAM) that contains instructions;

a memory cache that caches certain of said instructions from said DRAM; and

a tree engine that parses data within said PDUs and employs said DRAM and said memory cache to obtain ones of said instructions, including:

a multi-thread execution pipeline loop having a pipeline latency, and

a context switching system for said multi-thread execution pipeline loop, having:

a miss fulfillment first-in-first-out buffer (FIFO);

a context switch requesting subsystem that:

detects a device request from a thread executing within said multi-thread execution pipeline loop for access to a device having a fulfillment latency exceeding said pipeline latency, and

generates a context switch request for said thread, and

a context controller subsystem that receives said context switch request and, based thereon, stores said thread in said miss fulfillment FIFO until said device request is fulfilled, said thread sequencing through said entire miss fulfillment FIFO before exiting therefrom at a rate having a period associated with said pipeline latency.

16. (previously presented) The fast pattern processor as recited in Claim 15 wherein said context controller subsystem further allows a new thread to enter said multi-thread execution pipeline loop after said thread is stored in said FIFO.

17. (original) The fast pattern processor as recited in Claim 15 wherein said context controller subsystem further allows other threads within said multi-thread execution pipeline loop to continue to execute while said thread is waiting for said device request to be fulfilled.

18. (previously presented) The fast pattern processor as recited in Claim 15 wherein said

context switching system is further configured to:

store said thread in said miss fulfillment FIFO upon reaching an end position of said multi-thread execution pipeline loop, and

reinsert said thread into said multi-thread execution pipeline loop at a beginning position after said thread exits said miss fulfillment FIFO.

19. (previously presented) The fast pattern processor as recited in Claim 15 wherein said thread is looped back to a beginning stage of said multi-thread execution pipeline loop when said thread reaches an end stage of said multi-thread execution pipeline loop and said thread has not finished processing.

20. (currently amended) The fast pattern processor as recited in Claim 15 wherein said context controller subsystem sequences said thread through said miss fulfillment FIFO at a rate equal ~~having a period substantially equivalent~~ to said pipeline latency of said multi-thread execution pipeline loop.

21. (original) The fast pattern processor as recited in Claim 15 wherein said device is said DRAM and said device request is a request to access said DRAM due to a cache miss status from said memory cache.